Design of the STARS Network QoS Reservation System*

Gary Hoo, Keith Jackson, William Johnston

Abstract

We are developing a trusted advance reservation system for network bandwidth. By "trusted" we mean that the system will be fraud-resistant: it will authenticate and check the authorization of each service requestor, and will prevent man-in-the-middle attacks by digitally signing all information exchanged over open networks. The system will allow service requestors to reserve network bandwidth in advance of using it, in order to support, e.g., coordinated use of distributed computing resources such as remote instruments connected to supercomputers. In particular, the system is designed to control the use of *premium* bandwidth—that is, classes of service that are better than the default best-effort *quality of service* (QoS).

I. Introduction

In the near future, computationally intensive problems will be tackled by heterogeneous, distributed scientific and computing resources that are part of highly networked *computational and data grids* [7, 14]. Grid applications will often generate large quantities of data for real-time consumption by analysis, visualization, and/or storage devices elsewhere in the Grid. To attain the required high data transfer rates at the required time in shared networks, this traffic will have to be given preferential treatment. Better-than-best-effort network traffic classes have been defined by the Internet Engineering Task Force (IETF) and in ATM technology, among others. Such enhanced, or *premium*, QoS offerings will have to be administered so as to maintain the service guarantees of the QoS classes. Thus premium network bandwidth of all types will be a Grid resource no more or less important than any other resource, such as computing systems, when these resources must operate in concert to accomplish their task.

To ensure that Grid resources are available when they are needed, their use will have to be *reserved*, or scheduled in advance. While many scheduling and control systems exist for computational resources such as supercomputers, we are aware of no such systems for network bandwidth in shared IP networks. Furthermore, "the network" often will actually be an internetwork of multiple administratively distinct networks. The owner of each network will want to make its own decisions on how that network's resources are allocated. This administrative autonomy might extend to subnetworks or even to individual routers, depending on how the network is organized.

Every user would want his traffic to be treated as premium rather than as best-effort, so like any other scarce resource, allocation and use of premium network bandwidth must only be according to a well-defined policy. It must not be possible to reserve or to use such bandwidth in violation of policy governing its allocation and use.

*This work is supported by the U. S. Dept. of Energy, Office of Science, Office of Advanced Scientific Computing Research, Mathematical, Information, and Computational Sciences office (http://www.er.doe.gov/production/octr/mics), under contract DE-AC03-76SF00098 with the University of California. The authors may be contacted at: gjhoo@lbl.gov, krjackson@lbl.gov, wejohnston@lbl.gov. This document is report LBNL-45039.

STARS is a distributed system that coordinates the autonomous decision-making required to reserve premium bandwidth from a data source to a data sink. STARS permits each network service provider, including the domain of an end user, to decide how to allocate the premium bandwidth under its control. These decisions collectively represent a commitment of premium bandwidth by all the affected parties—the end sites, plus all the intermediate networks—between a source and sink. Each decision will depend in part on the identity of the requestor, so STARS provides strong authentication of one participant to another. The decisions also must be communicated between the participants, which raises the possibility of message-tampering and other types of fraud; we have therefore paid special attention to making STARS communications resistant to garbling, intentional modification, and false attribution. Claiming of reservations is protected by authenticating the claimant against the reservation owner and then securely passing the flow specification to the network so the flow's packets can be marked for premium service.

II. Concepts underlying STARS

Broadly, STARS permits one to treat a network or internetwork as a single entity for scheduling purposes. A requestor tries to reserve some type and amount of premium bandwidth, over a discrete time period, between two endpoints (STARS does not support multicast flows). If STARS determines the request is permitted by the policy of the domain whence it originates (e.g., the requestor's home institution), the request will be promulgated through the networks which will have to carry the traffic (and thus which must be reserved), including the domain for which the traffic is destined. The result is a yes-or-no answer.

A number of concepts underlie the STARS approach; we discuss them below, then describe the STARS architecture.

II.A. What is being reserved?

STARS was designed around the assumption that there are *restriction points* in a network which cannot accommodate all the premium traffic that could traverse them and which therefore must be scheduled. (If no restriction points exist, then STARS is unnecessary for that network.) Typically, a restriction point would be a router in which multiple incoming flows could converge on a single output port. The network administrators for each autonomous system (AS) are responsible for identifying potential and actual restriction points; how to do so is beyond the scope of this paper, though the authors are also working with the engineering group of the Energy Sciences Network (ESNet)—the U.S. Department of Energy's production scientific network—to address this issue.

The ordered set of restriction points between a data source and a data sink is called the *reservation path*. A *reservation* is a hold on an amount of premium bandwidth of a particular type for some time period. In the context of a particular restriction point, the term "reservation" applies only to the bandwidth reserved at that restriction point; for STARS as a whole, "reservation" denotes the ordered set of restriction-point reservations along a reservation path. The latter is also called an *end-to-end reservation* in this paper.

Note that reservations are unidirectional: that is, a reservation applies only to data moving from a data source to a data sink. However, two communicating entities may each be a source and sink at various times during a conversation. For example, if host A transmits data to host B using TCP, B sends acknowledgements to A even if it never sends any data of its own. To STARS, A and B are

both data sources, each with a reservation path. Distinguishing between these paths, though they both have the same endpoints, is important if routing is asymmetric or if premium bandwidth is scarce: TCP throughput, for example, is reduced if acknowledgements are delayed. Bidirectional data flows therefore require separate unidirectional reservations. (We are investigating how to automate reservations for "implicit" reverse flows, such as those of TCP acknowledgements.) STARS makes the two reservations for a bidirectional flow simultaneously.

Finally, we are aware that network QoS may be provided by any of several means. STARS can take advantage of whatever service differentiation mechanism—RSVP [2], DiffServ [1], MPLS [3], ATM QoS [6], etc.—is available in the network, because STARS is a framework on top of these mechanisms.

II.B. Domains and users

The domain whence a reservation request comes is the *originating domain*; by extension, the domain of the other endpoint in the request is the *destination domain*. One of STARS' main responsibilities is to decide whether or not a reservation may be made according to the policies governing bandwidth allocation in both domains. These policies may be expressed in terms of access control lists or other mechanisms tied to identity. For this reason, every STARS request must be associated with a user in the originating domain; this user (or his agent) is known as the *requestor.*

In the destination domain, the requestor's identity may not be recognized. (This situation is likely to arise in a Grid consisting of resources whose use is open to cooperating groups: cooperation may not extend to the onerous mutual issuance of login IDs *en masse*.) Nevertheless, the destination domain must be allowed to apply its policy rules to the consumption of its premium bandwidth along the "last mile" from the wide-area network to a host in that domain.

Since some process must consume the data on this host, one solution to this dilemma is to assign the rights of the owner of this process (presumably a cooperative collaborator of the requestor) to the requestor. The user (or his agent) associated with the consuming process is the *destination domain user.* The destination domain user delegates his rights to consume bandwidth in the destination domain to the requestor via a secure document called a *proxy credential*. During reservation, the requestor's identity and the proxy credential are presented to the destination domain, which uses this information to decide if the requestor may make the reservation based on the rights of the destination domain user.

II.C. Service-level agreement

STARS calls for active management of scarce network resources. Such management requires clearly defined policies drawn up by network administrators of all the autonomous systems (ASes) participating in the system, either as providers or users (or both). In particular, *service-level agreements* (SLAs) are required, defining the kind and amount of premium traffic an upstream domain may deliver to a downstream domain over a given time period. SLAs also describe the disposition of excess traffic (e.g., whether it is dropped or treated as best-effort), terms of payment, and the myriad other terms affecting the service provided by one domain to another. SLAs will govern STARS behavior during reservation as well as traffic treatment during

bandwidth use. SLAs are agreements between ASes and typically are not concerned with the identity of end users.

III. STARS architecture

STARS has three major components: reservation managers, the reservation coordinator, and the client-side reservation agent.

III.A. Reservation manager (RM)

A *reservation manager* (RM) is associated with each restriction point. The RM implements the advance reservation and security capabilities required to participate in STARS, essentially acting as the restriction point's proxy; this removes the need to modify the restriction point itself.

STARS advance reservation capabilities include creating, modifying, claiming, and removing reservations. In particular, the RM encapsulates a scheduler that understands advance reservation (e.g., a time slot-based system as opposed to a batch scheduler). The current implementation provides a *slot scheduler* [10, 9] that is used by default, but we are investigating how to use existing schedulers such as Maui [15, 4] or DSRT [5] instead so sites may use whatever scheduler is appropriate.

An RM may authenticate and perform access control checks on a requestor. It coordinates the scheduling of the restriction point(s) under its control during reservation, and communicates with the restriction point(s) to allow claiming of reservations.

An RM emits a digitally signed document called a *token* which represents the result of its reservation process. The result may be "success" or "failure." RMs *adjacent to* (directly upstream or downstream from) one another in the reservation path have trust relationships with one another that permit verification of tokens. Such verification (via digital signatures) is needed to prevent fraudulent reservations from being made by third parties impersonating valid RMs.

In addition, the RM can respond to *resource availability queries*. A resource availability query requests information about when and how much premium bandwidth of a given type is available within a given period. Such queries are particularly intended to assist high-level resource brokers that will negotiate on the requestor's behalf for the computational, storage, network, and other distributed resources the requestor will need to perform its task. Whether the RM can respond usefully to a resource availability query depends on its underlying scheduler(s), but the RM can at least return an "operation unsupported" or "information unavailable" response to the querier.

Note that an RM may represent more than one real restriction point. That is, an entire AS may look like a single restriction point because a different reservation scheme is used in the trusted interior of the network. RMs, with their relatively heavyweight security features, might only be used at the AS ingress points.

III.B. Reservation coordinator (RC)

To make an end-to-end reservation, a *requestor*, a person or a resource broker, must make a reservation at each RM in the reservation path. To do so, the requestor provides the addresses of the two communicating endpoints, the type of premium service desired, and the desired start and end

times to the *reservation coordinator* (RC). The RC authenticates the requestor, determines the reservation path, then contacts each RM in the reservation path(s) to make the reservation request. This step-by-step process continues until either an RM cannot satisfy the reservation request, or all RMs in the reservation path(s) have made the appropriate reservation(s).

The RC, given the two communicating endpoints, first must find the path between them in each direction, then determine which elements in that path are restriction points (i.e., find the reservation path). The client-side reservation agent (see Section III.C) is likely to participate in the path discovery, but this will not require any interaction with the requestor.

The RC contacts the RM associated with each restriction point and requests a reservation. The RC checks whether the returned token indicates a successful reservation was made. If so, the RC passes the token it just obtained to the RM representing the next restriction point in the reservation path (and saves the tokens in a secure repository of its own.) A "failure" token at any point terminates the reservation process, requiring the RC explicitly to cancel the reservations it has already made along the reservation path.

Note that in order for the RC to cancel a reservation autonomously—that is, without explicit action by the requestor—it must be trusted by the requestor. The RC acts as the requestor's proxy in this regard.

III.C. Client-side reservation agent (CRAg)

In the future, applications may be written to interact with STARS via a well-defined API. However, not all old applications can be modified to do so. STARS therefore provides the *client-side reservation agent* (CRAg), which acts as the requestor's proxy to STARS. In particular, the CRAg is meant to handle two tasks common to all applications: authentication and path discovery.

The requestor must authenticate itself to the RC or the RC would be vulnerable to replay attacks in which a malicious third party masqueraded as a legitimate user by passing old messages to the system. The CRAg obviates the need for every application to write its own authentication-challenge code.

If path discovery is implemented via, e.g., `traceroute` for an IP network, some entity on the host from which the data will flow must perform the `traceroute`. Again, it should not be necessary for every application to implement this functionality: rather, a commonly available module should do so.

The CRAg can be invoked as a standalone application which can make a STARS reservation, or can be called from within an application as a library module.

IV. Reservation

In the following sections we describe the reservation process itself, which is a two-stage commit process reflecting the need to permit network administrators to make their decisions independent of one another or of any central authority.

IV.A. Path and restriction point discovery

The first step in the reservation process is to locate each restriction point in the path between the end nodes. In our initial implementation, the RC starts by asking a CRAg to perform an IP `traceroute`, producing a list of routers between the endpoints. Next, the RC must determine which of these routers are restriction points. For our initial implementation, we plan that each AS should maintain a list of its routers which are RM-managed restriction points; how the AS determines these is at its discretion and beyond the scope of this paper. (In the future, the list might be generated dynamically according to network monitoring information.) The list also names the RM associated with each restriction point. The net result is a list of RMs the RC must contact to make the reservation—the reservation path.

The simple path discovery mechanism will not work if any AS in the path has disabled support for `traceroute`. We are investigating alternative methods in consultation with ESNet.

IV.B. End-to-end reservation

Having identified the restriction points in the reservation path, the RC starts making soft reservations on every RM in the path. A *soft reservation* is one which will be released after a short time if it has been neither refreshed nor upgraded to a *hard* (confirmed) reservation. The RC begins by contacting the RM associated with the originating node in the reservation path (the *soft reservation startpoint*): we call this the *first-hop RM*.

The first hop RM must authenticate the requestor and verify its right to make a reservation. We expect that authentication will be via X.509 certificates [11]. Authentication may be skipped if the RC indicates that authentication has already been performed by another trusted entity. This prevents duplication of effort if STARS is operating within a trusted domain using, e.g., the Globus Security Infrastructure [8].

The first-hop RM typically represents the local domain (site) of the requestor and is therefore responsible for implementing the policy of the requestor's site. Following authentication, an RM may apply access control by consulting a policy service. We are using the Akenti policy engine [17] for this purpose. The policy service will return "yes," "no," or "maybe," thus granting, denying, or deferring the decision on access (that is, the right to commit resources as requested). In general, a deferred decision means that the policy engine cannot evaluate certain policy conditions, perhaps because they depend on specific knowledge of the resource or are too variable to be encoded as static policy. The site RM is responsible for taking appropriate action in response to a deferred decision.

Following a "yes" from the policy service, or satisfaction of whatever ancillary conditions were required as part of a deferred decision, the request is passed to the appropriate scheduler. There may be multiple schedulers controlled by a single RM, each scheduler corresponding to a particular class of premium bandwidth at a particular restriction point. The RM requires that at minimum a scheduler must return either "yes" or "no," signifying success or failure in accommodating the reservation request. The RM treats a successful return as a soft reservation—that is, the requested resources are conditionally held for a short period of time. (This period will almost certainly be configurable and perhaps dynamically adjustable.) If the soft reservation is not upgraded within that time, it is automatically removed by the RM.

The RM stores the soft reservation in persistent (local) storage, then encodes the original reservation request information along with RM-specific information as an XML [18, 19] document. The RM digitally signs [16] the document, and the resulting *soft reservation token* is returned to the caller.

The other RMs in the reservation path each perform operations similar to the first-hop RM. However, policy-based access control based on the original requestor's identity will seldom be possible because the requestor's identity will seldom be meaningful to intermediate RMs. Instead, both authentication and authorization are based on the identity of the upstream (prior) RM in the reservation path. A (signed) soft reservation token indicates that the upstream RM has committed its resources, which is proof enough that the (original) request was genuine. While this places a great burden on the first-hop RM to perform stringent, fine-grained policy-checking, that is the appropriate place for the burden. At the same time, other RMs need not abandon policy-checking altogether: we envision that at the very least, RMs for ingress points to an autonomous system will verify that the reservation request is within the agreed-upon service limits described in the SLA negotiated with the upstream domain.

Following signature verification of the upstream RM's soft reservation token and any desired policy checking, the RM will extract the relevant reservation information from the token, schedule the request, create its own soft reservation token, and return that token to the RC.

At the RM associated with the destination node on the reservation path, or the *soft reservation endpoint*, the *last-hop RM* authenticates the identity of the destination domain user (see Section II.B), and checks the destination domain user's authority to make such a reservation. Again, we are using Akenti for authorization checking; however, the destination domain user will be identified by means of proxy credentials. The proxy credentials not only identify the destination domain user, but also securely associate the requestor's identity (and thus his rights) with the destination domain user's identity (and his rights) so the credentials cannot be used by an unauthorized party.

A failure of any RM to create the requested reservation is denoted by that RM returning a digitally signed *failure token*. Receipt of a failure token terminates the reservation process. Any existing soft reservations in the reservation path are either explicitly removed, or allowed to time out. (The RC removes all soft reservations along the reservation path on which the error occurred, but allows any soft reservations on the corresponding path in the opposite direction to time out. Explicit cancellation of the latter is difficult because there may not be a trust relationship between the node at which the failure occurred and the node at which the corresponding opposite path last successfully obtained a soft reservation before the failure.)

The RC refreshes outstanding soft reservations at all RMs until either the entire reservation process is complete or a failure has occurred.

IV.C. Soft-to-hard reservation upgrade process

The reservation upgrade process changes soft reservations to hard reservations. It proceeds from the last-hop RM backwards to the first-hop RM.

The RC initiates the upgrade process by sending an upgrade message to the last-hop RM. The upgrade message contains, among other things, the last-hop RM's soft reservation token, which

permits the last-hop RM to verify the validity of the request. The last-hop RM upgrades the soft reservation to a hard reservation internally and returns a hard reservation token to the RC. Like the soft reservation token, the hard reservation token is digitally signed.

Upon receiving this hard reservation token, the RC incorporates it into an upgrade message and sends that message to the immediately prior RM in the reservation path. That RM verifies the signature of the last-hop RM's hard reservation token, then upgrades its own soft reservation and returns a hard reservation token to the RC. In this way, the RC works backwards along the reservation path to the first-hop RM.

When making related unidirectional end-to-end reservations for a bidirectional flow, the RC waits until both directions have completed the soft reservation process before starting the upgrade to hard reservations.

IV.D. Claiming

When the requestor or its designated agent—hereinafter referred to as the *claimant*—requests that the reservation be instantiated, it presents its identity, the hard reservation token, and the flow specification of the process that will actually make the connection, to the first-hop RM. The first-hop RM authenticates the claimant and verifies the signature of the hard reservation token. (The token is not really necessary except insofar as it uniquely identifies the reservation being claimed. Furthermore, possession of the token by itself is insufficient to claim the reservation; the identity of the claimant must match the identity of the party entitled to claim the reservation, securely encoded within the token.) The first-hop RM also performs any runtime checking, e.g., "Is the claimant coming from the right subnet?" or "Has the state of the network changed so as to invalidate the reservation?" If the authorization checks succeed, the RM marks the reservation as having been claimed and returns a "success" message to the claimant. The RM passes the flow specification to the QoS manager in order that, in the case of IP DiffServ for example, the packets can be marked for the premium class.

The claimant also contacts the last-hop RM and tells it to claim the reservation; the process is similar to claiming at the first-hop RM. When both RMs have returned "success" messages, the claimant may begin sending data.

Intermediate RMs do not need to participate in the claiming process because their decisions are typically made based on the aggregate use of premium bandwidth being within the SLA between ASes.

The first-hop RM prepares a digitally signed *claim token* for accounting purposes, to signify that the reservation was claimed. Whether this should be kept solely by the RM, issued to the claimant, or published in an information service is under investigation.

V. Other work

Related work in bandwidth partitioning and reservation is being pursued by the Globus project and the Internet2 Quality of Service Backbone (I2 QBone) testbed.

The Globus Architecture for Reservation and Allocation (GARA) [9] defines a QoS architecture that supports flow-specific QoS specification as well as advance reservation. GARA developers

have conducted tests of their reservation framework in the wide area using a testbed that includes Lawrence Berkeley National Laboratory and Argonne National Laboratory. Because of the close relationship of our work to GARA, we are collaborating with the GARA developers to share code and experiences.

The I2 QBone [12] has been active in the design and deployment of an interdomain QoS framework. In particular, the I2 QBone Bandwidth Broker Advisory Council (BBAC) [13] has attempted both to identify the many administrative difficulties in propagating premium traffic between autonomous systems (ASes), and to design a protocol to address these difficulties. This work has focused mainly on developing *bandwidth brokers*, software entities that manage premium bandwidth for an AS and that can negotiate bandwidth requests with one another on behalf of end users using the prototype protocol. We are monitoring the BBAC's activities to determine if STARS can be made interoperable with a bandwidth broker, but at present believe it would only be possible if we disabled much of the STARS security framework.

VI. Current status

We have designed the RM and RC as well as the structure of the XML documents they will exchange. We have implemented significant portions of the RM and are implementing the RC. We are working with an ESNet QoS testbed in order to allow wide-area testing of the STARS approach.

VII. Conclusion

STARS is a trusted advance reservation system for premium network bandwidth. The system will authenticate and check the authorization of each requestor, and will prevent man-in-the-middle attacks by digitally signing all information exchanged over open networks. The system will allow service requestors to reserve network bandwidth in advance of using it, to support coordinated use of distributed computing resources such as supercomputers. The evolution of STARS is in no small part the result of lengthy discussions with ESNet engineers in order to ensure that the process can conform to the operational requirements of an ISP.

VIII. References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss. An Architecture for Differentiated Services. Internet Engineering Task Force Request for Comments 2475.

[2] R. Braden, ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification. Internet Engineering Task Force Request for Comments 2205.

[3] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, and A. Viswanathan. A Framework for Multiprotocol Label Switching. Internet Engineering Task Force Internet Draft (work in progress).

[4] C. Chisolm, L. Gerner, D. Jackson. Maui Scheduler. White paper available at http://pipeline.mhpcc.edu/research/mi97/97mi02.html.

[5] H. Chu, K. Nahrstedt, J. Qian. Dynamic Soft Real Time Scheduling Framework for Multimedia Environment. White paper available from http://monet.cs.uiuc.edu/dsrt2/.

[6] P. Ferguson and G. Huston. *Quality of Service: Delivering QoS on the Internet and in Corporate Networks*. John Wiley & Sons, Inc., 1998.

[7] I. Foster and C. Kesselman, eds. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, Inc., 1999.

[8] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. "A Security Architecture for Computational Grids." *Proc. 5th ACM Conference on Computer and Communications Security Conference*, pg. 83-92, 1998.

[9] I. Foster, A. Roy, V. Sander, and L. Winkler. "End-to-End Quality of Service for High-End Applications." Submitted to *IEEE Journal on Selected Areas in Communications Special Issue on QoS in the Internet*, 1999.

[10] G. Hoo, W. Johnston, I. Foster, and A. Roy. QoS as Middleware: Bandwidth Brokering System Design. Lawrence Berkeley National Laboratory technical report LBNL-42947.

[11] R. Housley, W. Ford, W. Polk, D. Solo. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. Internet Engineering Task Force Request for Comments 2459.

[12] Internet2 Quality of Service Backbone home page. http://www.internet2.edu/qos/qbone/.

[13] Internet2 QBone Bandwidth Broker Advisory Council home page. http://www.internet2.edu/qos/qbone/QBBAC.shtml.

[14] W. Johnston, D. Gannon and B. Nitzberg. "Grids as Production Computing Environments: The Engineering Aspects of NASA's Information Power Grid." *Proc. of the Eighth IEEE International Symposium on High Performance Distributed Computing*, Redondo Beach, CA, August 3-6, 1999.

[15] Maui Scheduler home page. http://pipeline.mhpcc.edu/maui/.

[16] J. Reagle. XML-Signature Requirements. Internet Engineering Task Force Internet Draft/ World Wide Web Consortium Working Draft (work in progress; production of the joint IETF/ W3C XML Signature Working Group).

[17] M. Thompson, W. Johnston, S. Mudumbai, G. Hoo, K. Jackson, A. Essiari. "Certificate-based Access Control for Widely Distributed Resources." *Proc. of the Eighth Usenix Security Symposium*, August 23-26, 1999.

[18] World Wide Web Consortium. The Extensible Markup Language (XML) home page. http://www.w3.org/XML/.

[19] World Wide Web Consortium. Extensible Markup Language (XML) 1.0. http://www.w3.org/TR/1998/REC-xml-19980210.